🎮 GameHub - Tutorial: Sistema de Comunidade

📚 Índice

---

1. Entendendo a Estrutura Atual

📁 O que você já tem:

```
app/
├── _layout.tsx          ← Sistema de temas e navegação
├── +not-found.tsx        ← Página de erro 404
├── beltranis.tsx        ← Perfil do player
├── xulambs.tsx           ← Demo de componentes
├── constants/
│   └── theme.ts          ← Temas e estilos
└── contexts/
    └── ThemeContext.tsx ← Contexto de temas
```

🎨 Sistema de Temas

Você já tem temas prontos que podem ser trocados dinamicamente:

- **Retro**: Estilo cyberpunk/neon
- **Minimal**: Estilo limpo e moderno
- **Dark**: Escuro profundo
- **Light**: Claro e clean

Como funciona:

typescript

```
// ThemeContext fornece:
themeName    // Nome do tema atual
changeTheme  // Função para trocar tema
```

---

2. Conceitos Importantes

🔗 Links Úteis para Estudar:

TypeScript Basics

- TypeScript Handbook
- O que são tipos e interfaces
- Por que usar TypeScript no React Native

React Hooks

- useState - Gerenciar estados
- useContext - Acessar contextos
- useEffect - Efeitos colaterais

Styled Components

- Styled Components Native
- Como estilizar componentes
- Props dinâmicas em estilos

React Native Components

- ScrollView - Lista com scroll
- FlatList - Lista otimizada
- TouchableOpacity - Botões tocáveis

---

3. Planejamento da Comunidade

🎯 O que vamos construir:

```
COMUNIDADE
├── Feed de Posts
│   ├── Avatar do usuário
│   ├── Nome e informações
│   ├── Conteúdo do post
│   ├── Imagem (opcional)
│   ├── Botões: Like, Comment, Share
│   └── Contador de interações
├── Formulário para Novo Post
│   ├── Campo de texto
│   └── Botão publicar
└── Sistema de Comentários (bonus)
```

📊 Estrutura de Dados:

typescript

```typescript
Post = {
  id: string
  author: { name, avatar, level }
  content: string
  image?: string
  timestamp: Date
  likes: number
  comments: number
  shares: number
}
```

Passo 1: Criar Tipos TypeScript

📝 Criar arquivo `types/community.ts`

typescript

```typescript
// types/community.ts

export interface User {
  id: string;
  name: string;
  avatar: string; // Iniciais para avatar
  level: number;
  rank: string; // "Bronze", "Silver", "Gold", etc.
}

export interface Post {
  id: string;
  author: User;
  content: string;
  image?: string; // URL da imagem (opcional)
  timestamp: Date;
  likes: number;
  comments: number;
  shares: number;
  isLiked: boolean; // Se o usuário atual curtiu
}

export interface Comment {
  id: string;
  author: User;
  content: string;
  timestamp: Date;
}
```

Por que fazer isso?

- TypeScript ajuda a evitar erros
- Autocomplete no editor
- Documentação automática do código

📚 **Estude:** TypeScript Interfaces

Passo 2: Criar Componente de Post

🎨 Criar arquivo `components/PostCard.tsx`

📋

typescript

```tsx
// components/PostCard.tsx

import React from "react";
import { View, Alert } from "react-native";
import styled, { DefaultTheme } from "styled-components/native";
import { spacing, typography } from "../constants/theme";
import { Post } from "../types/community";

// ============================================
// PROPS DO COMPONENTE
// ============================================
interface PostCardProps {
  post: Post;
  onLike: (postId: string) => void;
  onComment: (postId: string) => void;
  onShare: (postId: string) => void;
}

// ============================================
// COMPONENTE PRINCIPAL
// ============================================
export const PostCard: React.FC<PostCardProps> = ({
  post,
  onLike,
  onComment,
  onShare,
}) => {
  // Formatar tempo relativo (ex: "há 2 horas")
  const getTimeAgo = (date: Date): string => {
    const seconds = Math.floor((new Date().getTime() - date.getTime()) / 1000);

    if (seconds < 60) return "agora";
    if (seconds < 3600) return `${Math.floor(seconds / 60)}min`;
    if (seconds < 86400) return `${Math.floor(seconds / 3600)}h`;
    return `${Math.floor(seconds / 86400)}d`;
  };

  return (
    <Card>
      {/* HEADER DO POST */}
      <Header>
        <Avatar>
          <AvatarText>{post.author.avatar}</AvatarText>
        </Avatar>
    <UserInfo>
      <UserName>{post.author.name}</UserName>
      <MetaInfo>
        Lvl {post.author.level} • {post.author.rank} • {getTimeAgo(post.timestamp)}
      </MetaInfo>
    </UserInfo>
  </Header>

  {/* CONTEÚDO DO POST */}
  <Content>{post.content}</Content>

  {/* IMAGEM (SE HOUVER) */}
  {post.image && (
    <PostImage source={{ uri: post.image }} resizeMode="cover" />
  )}

  {/* ESTATÍSTICAS */}
  <Stats>
    <StatText>{post.likes} likes</StatText>
    <StatText>•</StatText>
    <StatText>{post.comments} comentários</StatText>
    <StatText>•</StatText>
    <StatText>{post.shares} compartilhamentos</StatText>
  </Stats>

  {/* BOTÕES DE AÇÃO */}
  <Actions>
    <ActionButton onPress={() => onLike(post.id)}>
      <ActionIcon>{post.isLiked ? "❤️" : "🤍"}</ActionIcon>
      <ActionText>Like</ActionText>
    </ActionButton>

    <ActionButton onPress={() => onComment(post.id)}>
      <ActionIcon>💬</ActionIcon>
      <ActionText>Comentar</ActionText>
    </ActionButton>

    <ActionButton onPress={() => onShare(post.id)}>
      <ActionIcon>🔗</ActionIcon>
      <ActionText>Compartilhar</ActionText>
    </ActionButton>
  </Actions>
</Card>
```

```
  );
};

// ==========================================
// COMPONENTES ESTILIZADOS
// ==========================================

const Card = styled.View`
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.surface};
  border-radius: 15px;
  padding: ${spacing.md}px;
  margin-bottom: ${spacing.md}px;
  border: 1px solid ${({ theme }: { theme: DefaultTheme }) => theme.border};
`;

const Header = styled.View`
  flex-direction: row;
  align-items: center;
  margin-bottom: ${spacing.md}px;
`;

const Avatar = styled.View`
  width: 45px;
  height: 45px;
  border-radius: 22.5px;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.primary};
  justify-content: center;
  align-items: center;
  margin-right: ${spacing.sm}px;
`;

const AvatarText = styled.Text`
  color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  font-size: ${typography.sizes.md}px;
  font-weight: 600;
`;

const UserInfo = styled.View`
  flex: 1;
`;

const UserName = styled.Text`
  font-size: ${typography.sizes.md}px;
  font-weight: 600;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
`;

const MetaInfo = styled.Text`
  font-size: ${typography.sizes.xs}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.textMuted};
  margin-top: 2px;
`;

const Content = styled.Text`
  font-size: ${typography.sizes.md}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
  line-height: 22px;
  margin-bottom: ${spacing.md}px;
`;

const PostImage = styled.Image`
  width: 100%;
  height: 200px;
  border-radius: 12px;
  margin-bottom: ${spacing.md}px;
`;

const Stats = styled.View`
  flex-direction: row;
  gap: ${spacing.xs}px;
  padding-bottom: ${spacing.sm}px;
  border-bottom-width: 1px;
  border-bottom-color: ${({ theme }: { theme: DefaultTheme }) => theme.border};
  margin-bottom: ${spacing.sm}px;
`;

const StatText = styled.Text`
  font-size: ${typography.sizes.xs}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.textMuted};
`;

const Actions = styled.View`
  flex-direction: row;
  justify-content: space-around;
`;

const ActionButton = styled.TouchableOpacity`
  flex-direction: row;
  align-items: center;
  gap: ${spacing.xs}px;
  padding: ${spacing.sm}px;
`;

const ActionIcon = styled.Text`
  font-size: 20px;
`;

const ActionText = styled.Text`
  font-size: ${typography.sizes.sm}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.textSecondary};
  font-weight: 600;
`;
```

O que fizemos:

1. ✅ Criamos interface para as props
2. ✅ Componente recebe um post e callbacks
3. ✅ Exibe avatar, nome, conteúdo, imagem
4. ✅ Botões interativos (Like, Comment, Share)
5. ✅ Estilos usando o tema dinâmico

📚 **Estude:** React Components

Passo 3: Tela de Comunidade

📱 Criar arquivo `community.tsx`

typescript

```tsx
// community.tsx

import React, { useState } from "react";
import { Alert, KeyboardAvoidingView, Platform } from "react-native";
import styled, { DefaultTheme } from "styled-components/native";
import { spacing, typography } from "./constants/theme";
import { Post } from "./types/community";
import { PostCard } from "./components/PostCard";

// ============================================
// DADOS MOCKADOS (simulados)
// ============================================
const MOCK_POSTS: Post[] = [
  {
    id: "1",
    author: {
      id: "user1",
      name: "ShadowGamer",
      avatar: "SG",
      level: 85,
      rank: "Diamante",
    },
    content: "Acabei de zerar Dark Souls pela 10ª vez! Quem mais é viciado nesse jogo? 🎮 🔥 ",
    timestamp: new Date(Date.now() - 2 * 60 * 60 * 1000), // 2h atrás
    likes: 42,
    comments: 8,
    shares: 3,
    isLiked: false,
  },
  {
    id: "2",
    author: {
      id: "user2",
      name: "ProGamer99",
      avatar: "PG",
      level: 92,
      rank: "Mestre",
    },
    content: "Alguém quer jogar Valorant agora? Preciso de um time pra ranked!",
    image: "https://picsum.photos/400/300?random=1",
    timestamp: new Date(Date.now() - 5 * 60 * 60 * 1000), // 5h atrás
    likes: 15,
    comments: 12,
    shares: 1,
    isLiked: true,
  },
  {
    id: "3",
    author: {
      id: "user3",
      name: "NinjaKiller",
      avatar: "NK",
      level: 73,
      rank: "Platina",
    },
    content: "Setup novo chegou! RTX 4090 + i9-13900K. Agora sim vou dominar! 💻 ⚡ ",
    image: "https://picsum.photos/400/300?random=2",
    timestamp: new Date(Date.now() - 24 * 60 * 60 * 1000), // 1 dia atrás
    likes: 128,
    comments: 34,
    shares: 12,
    isLiked: false,
  },
];

// ============================================
// COMPONENTE PRINCIPAL
// ============================================
export default function CommunityScreen() {
  const [posts, setPosts] = useState<Post[]>(MOCK_POSTS);
  const [newPostText, setNewPostText] = useState("");

  // ========== HANDLERS ==========

  const handleLike = (postId: string) => {
    setPosts((prevPosts) =>
      prevPosts.map((post) =>
        post.id === postId
          ? {
              ...post,
              likes: post.isLiked ? post.likes - 1 : post.likes + 1,
              isLiked: !post.isLiked,
            }
          : post
      )
    );
  };
```

```
const handleComment = (postId: string) => {
  Alert.alert("Comentar", `Abrindo comentários do post ${postId}`);
};

const handleShare = (postId: string) => {
  setPosts((prevPosts) =>
    prevPosts.map((post) =>
      post.id === postId
        ? { ...post, shares: post.shares + 1 }
        : post
    )
  );
  Alert.alert("Compartilhado!", "Post compartilhado com sucesso!");
};

const handlePublishPost = () => {
  if (newPostText.trim() === "") {
    Alert.alert("Erro", "Digite algo antes de publicar!");
    return;
  }

  const newPost: Post = {
    id: Date.now().toString(),
    author: {
      id: "currentUser",
      name: "Você",
      avatar: "VC",
      level: 50,
      rank: "Ouro",
    },
    content: newPostText,
    timestamp: new Date(),
    likes: 0,
    comments: 0,
    shares: 0,
    isLiked: false,
  };

  setPosts([newPost, ...posts]); // Adiciona no início
  setNewPostText(""); // Limpa o campo
  Alert.alert("Sucesso!", "Post publicado na comunidade!");
};

return (
    <Container>
      {/* HEADER */}
      <Header>
        <HeaderTitle>🌐 Comunidade GameHub</HeaderTitle>
        <HeaderSubtitle>Conecte-se com outros gamers</HeaderSubtitle>
      </Header>

      {/* FORMULÁRIO NOVO POST */}
      <NewPostSection>
        <NewPostInput
          placeholder="Compartilhe algo com a comunidade..."
          placeholderTextColor="#888"
          multiline
          value={newPostText}
          onChangeText={setNewPostText}
          maxLength={500}
        />
        <PublishButton onPress={handlePublishPost}>
          <PublishButtonText>📤 PUBLICAR</PublishButtonText>
        </PublishButton>
      </NewPostSection>

      {/* FEED DE POSTS */}
      <FeedScrollView
        contentContainerStyle={{ paddingBottom: spacing.xl }}
        showsVerticalScrollIndicator={false}
      >
        {posts.map((post) => (
          <PostCard
            key={post.id}
            post={post}
            onLike={handleLike}
            onComment={handleComment}
            onShare={handleShare}
          />
        ))}
      </FeedScrollView>
    </Container>
  );
}

// ============================================
// COMPONENTES ESTILIZADOS
// ============================================
const Container = styled.View`
  flex: 1;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
`;

const Header = styled.View`
  padding: ${spacing.lg}px;
  padding-top: ${spacing.xl}px;
  border-bottom-width: 2px;
  border-bottom-color: ${({ theme }: { theme: DefaultTheme }) => theme.border};
  align-items: center;
`;

const HeaderTitle = styled.Text`
  font-size: ${typography.sizes.heading}px;
  font-weight: 700;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.primary};
  margin-bottom: ${spacing.xs}px;
`;

const HeaderSubtitle = styled.Text`
  font-size: ${typography.sizes.sm}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.textSecondary};
`;

const NewPostSection = styled.View`
  padding: ${spacing.md}px;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.surface};
  border-bottom-width: 1px;
  border-bottom-color: ${({ theme }: { theme: DefaultTheme }) => theme.border};
`;

const NewPostInput = styled.TextInput`
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  border-radius: 12px;
  padding: ${spacing.md}px;
  font-size: ${typography.sizes.md}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
  min-height: 80px;
  margin-bottom: ${spacing.sm}px;
  border: 1px solid ${({ theme }: { theme: DefaultTheme }) => theme.border};
`;
```

```
const PublishButton = styled.TouchableOpacity`
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.primary};
  padding: ${spacing.sm}px ${spacing.md}px;
  border-radius: 25px;
  align-items: center;
`;

const PublishButtonText = styled.Text`
  color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  font-size: ${typography.sizes.md}px;
  font-weight: 600;
`;

const FeedScrollView = styled.ScrollView`
  flex: 1;
  padding: ${spacing.md}px;
`;
```

O que fizemos:

1. ✅ Criamos dados mockados (simulados) para testar
2. ✅ Sistema de likes funcional
3. ✅ Formulário para criar novos posts
4. ✅ Feed scrollável com todos os posts
5. ✅ Callbacks para Like, Comment, Share

📚 **Estude:** Managing State

Passo 4: Sistema de Comentários

💬 Criar componente de comentários

📋✓

typescript

```tsx
// components/CommentSection.tsx

import React, { useState } from "react";
import styled, { DefaultTheme } from "styled-components/native";
import { spacing, typography } from "../constants/theme";
import { Comment } from "../types/community";

interface CommentSectionProps {
  postId: string;
  comments: Comment[];
  onAddComment: (postId: string, content: string) => void;
}

export const CommentSection: React.FC<CommentSectionProps> = ({
  postId,
  comments,
  onAddComment,
}) => {
  const [commentText, setCommentText] = useState("");

  const handleSubmit = () => {
    if (commentText.trim() === "") return;
    onAddComment(postId, commentText);
    setCommentText("");
  };

  return (
    <Container>
      <Title>💬 Comentários ({comments.length})</Title>

      {/* LISTA DE COMENTÁRIOS */}
      {comments.map((comment) => (
        <CommentCard key={comment.id}>
          <CommentAvatar>
            <AvatarText>{comment.author.avatar}</AvatarText>
          </CommentAvatar>
          <CommentContent>
            <CommentAuthor>{comment.author.name}</CommentAuthor>
            <CommentText>{comment.content}</CommentText>
          </CommentContent>
        </CommentCard>
      ))}

      {/* CAMPO PARA NOVO COMENTÁRIO */}
      <InputContainer>
        <CommentInput
          placeholder="Escreva um comentário..."
          value={commentText}
          onChangeText={setCommentText}
        />
        <SendButton onPress={handleSubmit}>
          <SendButtonText>➤</SendButtonText>
        </SendButton>
      </InputContainer>
    </Container>
  );
};

// Estilos...
const Container = styled.View`
  padding: ${spacing.md}px;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.surface};
`;

const Title = styled.Text`
  font-size: ${typography.sizes.lg}px;
  font-weight: 600;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
  margin-bottom: ${spacing.md}px;
`;

const CommentCard = styled.View`
  flex-direction: row;
  margin-bottom: ${spacing.md}px;
`;

const CommentAvatar = styled.View`
  width: 35px;
  height: 35px;
  border-radius: 17.5px;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.primary};
  justify-content: center;
  align-items: center;
  margin-right: ${spacing.sm}px;
`;

const AvatarText = styled.Text`
  color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  font-size: ${typography.sizes.sm}px;
  font-weight: 600;
`;

const CommentContent = styled.View`
  flex: 1;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  padding: ${spacing.sm}px;
  border-radius: 12px;
`;

const CommentAuthor = styled.Text`
  font-size: ${typography.sizes.sm}px;
  font-weight: 600;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
  margin-bottom: 4px;
`;

const CommentText = styled.Text`
  font-size: ${typography.sizes.sm}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.textSecondary};
`;

const InputContainer = styled.View`
  flex-direction: row;
  gap: ${spacing.sm}px;
`;

const CommentInput = styled.TextInput`
  flex: 1;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  border-radius: 20px;
  padding: ${spacing.sm}px ${spacing.md}px;
  font-size: ${typography.sizes.sm}px;
  color: ${({ theme }: { theme: DefaultTheme }) => theme.text};
`;

const SendButton = styled.TouchableOpacity`
  width: 40px;
  height: 40px;
  border-radius: 20px;
  background-color: ${({ theme }: { theme: DefaultTheme }) => theme.primary};
  justify-content: center;
  align-items: center;
```

```
`;

const SendButtonText = styled.Text`
  color: ${({ theme }: { theme: DefaultTheme }) => theme.background};
  font-size: 18px;
`;
```

Passo 5: Adicionar à Navegação

🗺️ Atualizar `_layout.tsx`

typescript

```typescript
// _layout.tsx (adicione esta linha)

<Stack>
  <Stack.Screen name="(tabs)" options={{ headerShown: false }} />
  <Stack.Screen name="+not-found" />
  <Stack.Screen name="xulambs" />
  <Stack.Screen name="beltranis" />
  <Stack.Screen
    name="community"
    options={{
      title: "Comunidade",
      headerShown: true
    }}
  />
</Stack>
```

🔗 Criar link para a comunidade

Em qualquer tela, adicione:

typescript

```
import { Link } from "expo-router";

<Link href="/community">
  <ButtonText>Ir para Comunidade</ButtonText>
</Link>
```

Melhorias e Próximos Passos

🚀 Funcionalidades Avançadas:

1. Persistência de Dados

typescript

```
import AsyncStorage from '@react-native-async-storage/async-storage';

// Salvar posts
await AsyncStorage.setItem('posts', JSON.stringify(posts));

// Carregar posts
const saved = await AsyncStorage.getItem('posts');
const posts = saved ? JSON.parse(saved) : [];
```

📚 **Estude:** AsyncStorage

2. Upload de Imagens

typescript

```
import * as ImagePicker from 'expo-image-picker';

const pickImage = async () => {
  const result = await ImagePicker.launchImageLibraryAsync({
    mediaTypes: ImagePicker.MediaTypeOptions.Images,
    allowsEditing: true,
    aspect: [4, 3],
    quality: 1,
  });

  if (!result.canceled) {
    setImage(result.assets[0].uri);
  }
};
```

📚 **Estude:** Expo Image Picker

3. API Backend Real

typescript

```
// Conectar com backend
const fetchPosts = async () => {
  const response = await fetch('https://api.gamehub.com/posts');
  const data = await response.json();
  setPosts(data);
};
```

📚 **Estude:** Fetch API

4. Notificações Push

typescript

```
import * as Notifications from 'expo-notifications';

// Quando alguém comenta no seu post
await Notifications.scheduleNotificationAsync({
  content: {
    title: "Novo comentário!",
    body: "Alguém comentou no seu post",
  },
  trigger: null,
});
```

📚 **Estude:** [Expo Notifications]

5. Filtros e Busca

typescript

```
const [filter, setFilter] = useState('all'); // 'all', 'liked', 'mine'

const filteredPosts = posts.filter(post => {
  if (filter === 'liked') return post.isLiked;
  if (filter === 'mine') return post.author.id === currentUserId;
  return true;
});
```

🎓 Checklist de Aprendizado

Básico:

- [ ] Entender TypeScript interfaces
- [ ] Usar useState para estados
- [ ] Criar componentes reutilizáveis
- [ ] Estilizar com styled-components
- [ ] Passar props entre componentes

Intermediário:

- [ ] Gerenciar arrays de objetos
- [ ] Implementar CRUD (Create, Read, Update, Delete)
- [ ] Usar useContext para temas
- [ ] Validação de formulários
- [ ] Tratamento de erros

Avançado:

- [ ] Integração com API
- [ ] Upload de imagens
- [ ] Persistência de dados local
- [ ] Otimização de performance
- [ ] Notificações push

📖 Recursos Adicionais

Documentação Oficial:

- React Native Docs
- Expo Docs
- TypeScript Docs

Tutoriais em Português:

- React Native - Rocketseat
- Curso React Native - YouTube

Comunidades:

- React Native Brasil - Discord
- [Stack Overflow em Português](https://pt.stackoverflow.com/questions/tagged/react-